LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

UCRL-TR-224345

Improving Interpolation in BoomerAMG

Josh Nolting, Ulrike Yang

September 11, 2006

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

Improving Interpolation in BoomerAMG

Josh Nolting
Ulrike Yang
LLNL/University of Colorado

Introduction

With new more aggressive coarsening algorithms that while reducing memory also degrade convergence often dramatically, it was imperative that new interpolation routines be implemented to recover this degradation. The implementation details and results for three new interpolation routines, standard, extended, and F-F, are presented in this paper. The project was focused on parallel implementation, so there is little theoretical analysis. The references contain much of the algorithmic design issues and analysis, if further understanding or exploration is needed, see [1][2][3]. It will be shown throughout this paper that long-range interpolation is needed for a number of problems, for some cases, there is almost a reduction in iterations by 2 orders of magnitude.

Background

Three new interpolation routines were implemented into BoomerAMG, all of which are based on long-range interpolation. Throughout this paper, F represents a fine point, C represents a coarse point, P_i represents the interpolatory set of point i, N_i represents the neighborhood of point i, and it will be assumed that $i \in F$. The standard interpolation routine will be discussed first, followed by extended, and finally F-F. For all of the schemes, there are some underlying assumptions; given a C/F-splitting and sets $P_i \subseteq C(i \in F)$ of interpolatory points, the goal is to define the interpolation weights w_{ik} in

$$e_i = \sum_{k \in P_i} w_{ik} e_k \ (i \in F) \qquad (1)$$

so that (1) yields a reasonable approximation for any algebraically smooth e which approximately satisfies*j*

$$a_{ii} e_i + \sum_{j \in N_i} a_{ij} e_j = 0 (i \in F) \ [1]. \qquad (2)$$

Because long range interpolation is used, for all of the new interpolation routines, the interpolatory set of a fine point *i* consists of strong coarse neighbor points as well as strong C-point connections of j, where $j \in F^s$ = {strong F neighbors of i}. Using this possibly bigger interpolatory set, the standard method is an extension of direct interpolation. For this method interpolation weights are generated by substituting the error e_j in (2) for all $j \in F^s$ using the *j*-th row of the matrix and applying direct interpolation to the obtained new equation. There are two ways to calculate the weights for standard interpolation. The negative and positive coefficients can either be calculated separately or together. It will be seen later, that this can have a large influence on the convergence. The extended method determines the weights analogously with classical interpolation using the extended interpolatory set, i.e. strong F connections are collapsed to the C points. With the extended interpolatory set, the points $j \in F^s$ are collapsed to the diagonal as well. Finally, F-F interpolation is just a modification to extended interpolation. The main difference is that the points $j \in F^s$ are not collapsed to the diagonal, and the interpolatory set is only extended if $j \in F^s$ has no common strong coarse neighbor with i.

Implementation

The ParCSR format is used to store the BoomerAMG matrices, with rows partitioned between the processors. The unknowns corresponding to the diagonal entry of these rows will be considered local to the processor. Each ParCSR matrix consists of two matrices, one containing the local unknowns and one containing the rest of the unknowns (or off- processor column non-zeros). More detailed discussions concerning the design and data structures of BoomerAMG can be found in [4]. With this information accessible, it is easy to get information for the neighbors of the unknown. However, with long-range interpolation, neighbors of neighbor's can possibly be used as interpolatory points and can affect interpolation weights. Thus, before determining the interpolation points and weights, the neighbors of neighbors information needs to be determined and made accessible. First, the full row information for the off-processor unknowns is gathered. New off processor points (neighbors of neighbors not previously seen) can then be

determined and off-processor arrays can be formed (C/F-splitting, fine-to-coarse mappings, column mapping for off-processor matrices, etc. [4]) Because of the monotonicity required in some of these arrays, care needs to be taken to correctly store the new off-processor points. Also, since new points could force communication with new processors, communication patterns need to be adjusted. With all the needed information at hand, the interpolation matrix can be formed. This is implemented with two main loops, both looping over the local fine points. The first loop is used to determine the size of the interpolation matrix, the second determines the weights and the column locations for the local interpolation matrix. After the 2^{nd} loop, the communication pattern for the interpolation matrix is determined and the column locations for the off-processor interpolation matrix are found and stored.

Results

The new interpolation routines are effective for situations in which coarsening leads to strong F - F connections with no common C-point. For all of the tests presented below, PMIS coarsening will be used [5]. Also, there are tools and schemes that could be used to control complexity or accelerate convergence, however, for these tests, no special treatment will be given so that each scheme can be studied as a standalone method. The 3-dimensional problems were conducted with processor counts equal to 1, 8, 64, and 512. The problem size was kept approximately at 40 X 40 X 40 unknowns on each processor. In other words, as the problem size doubles in each direction, the number of processors used grows by a factor of 8. For the 2-dimensional case, the problem size for each processor will be approximately 512 X 512 unknowns and processor counts will be equal to 1, 4, 16, 64, and 256. The PDEs will be discretized using finite difference methods. For these test cases, if the residual has not been reduced below the tolerance after 1000 iterations, it is considered not converging.

The 3d Poisson problem, with constant righthand side will be explored first. Classical and multipass are two interpolation routines that are already implemented, and they will be used for comparisons.

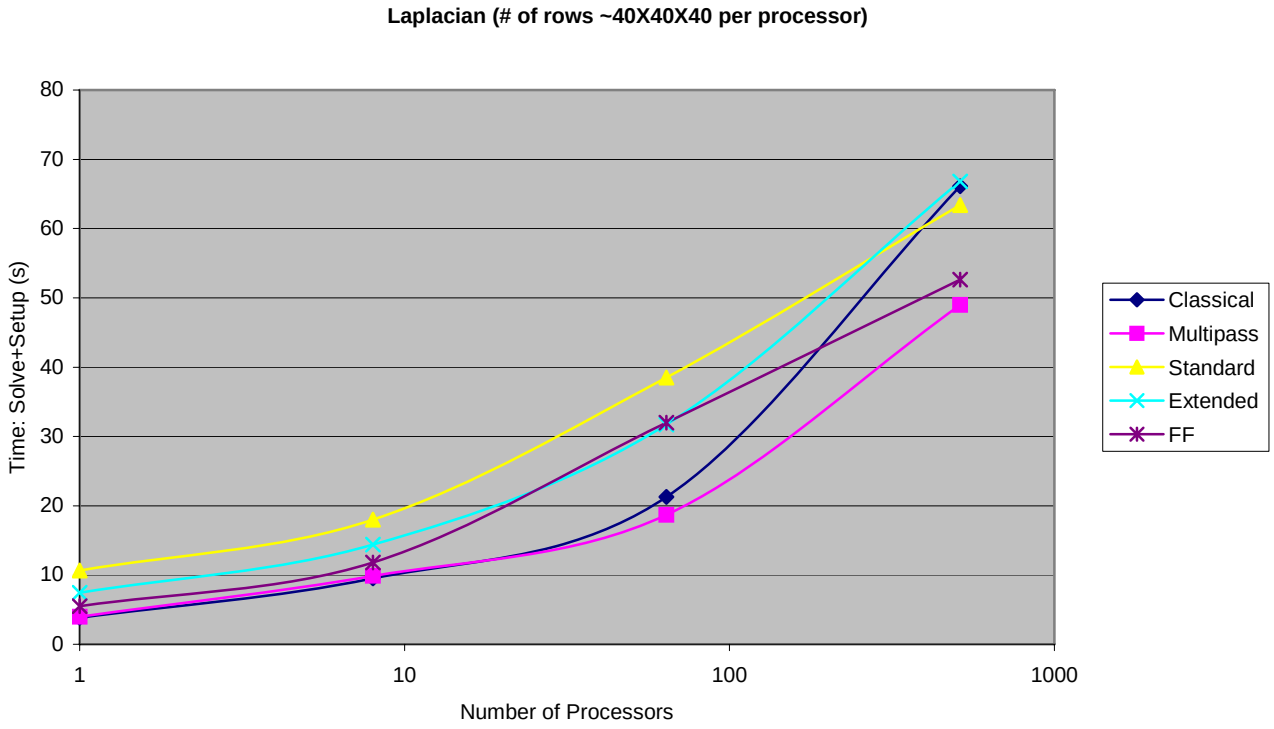

Figure 1: Time to needed to reduce the relative residual below 10^{-7}.

For small test cases, it appears that the original methods are satisfactory, but as the problem size increases, it starts to become more important to use the long-range interpolation routines. The long-range methods consistently need more time for setup, but less iterations. The results for the new routines could be improved greatly if the interpolation matrix was truncated. However, this test case still serves as a good base test.

A PDE with jumps will be used as the 2^{nd} test. This is a good problem, because PMIS coarsening could be used as a viable option for coarsening. It then becomes imperative to use long-range interpolation. The "classical" method does not even converge for the smallest problem so it is purged from the plot below. Also, the standard routine is not scaling at all, and needs 297 iterations to converge for the 64 processor case. This is not very good when compared to extended and F-F, which converged in 11 and 15 iterations respectively. For this problem it would be best to use extended or F-F, with F-F consistently using less time. The multipass routine is working for small problems but is beginning to struggle as the problem size increases.

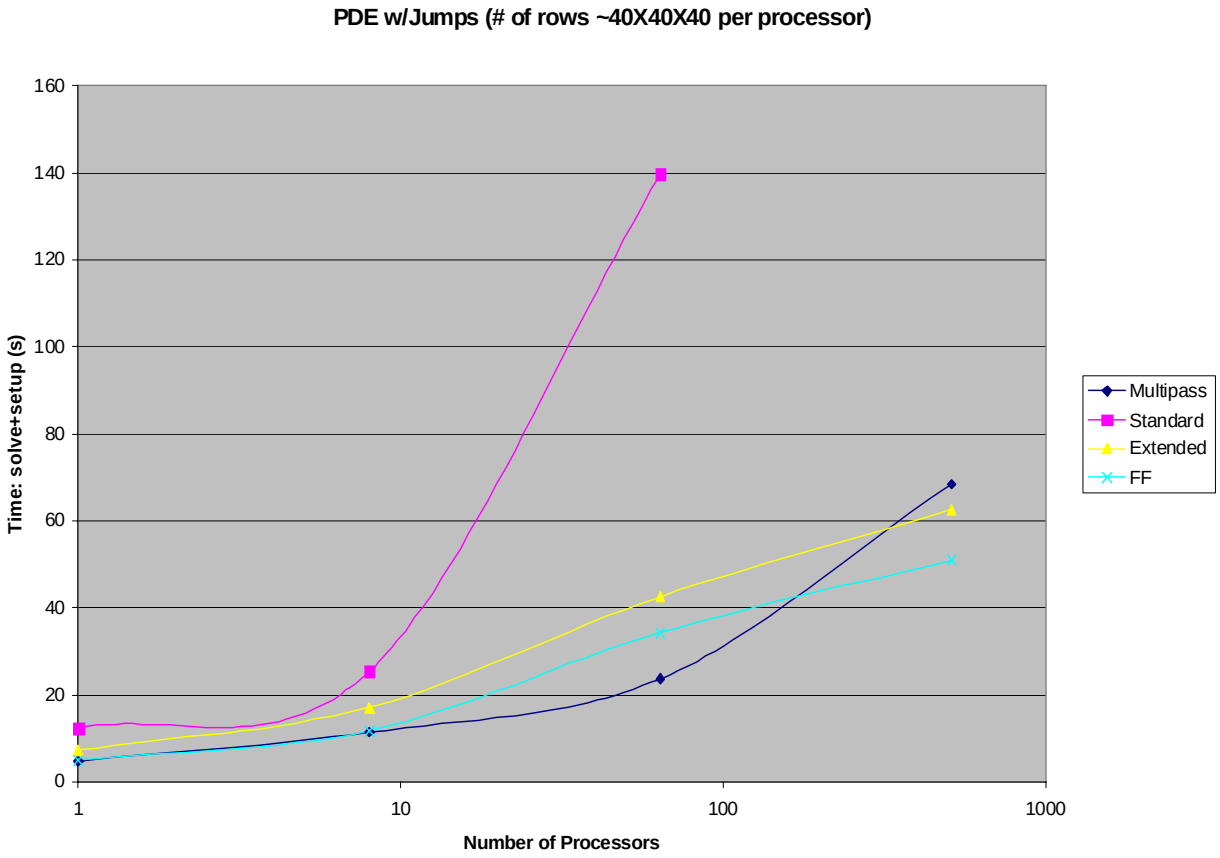

Figure 2: Time to needed to reduce the relative residual below 10^{-7}.

It can be seen below, that the setup times for multipass are very small and scaling very well, however, the iterations to converge are not scaling as well, causing the setup plus solve times to become greater than extended and F-F. It requires 14 and 17 iterations to converge for extended and F-F respectively, and 200 iterations for multipass.

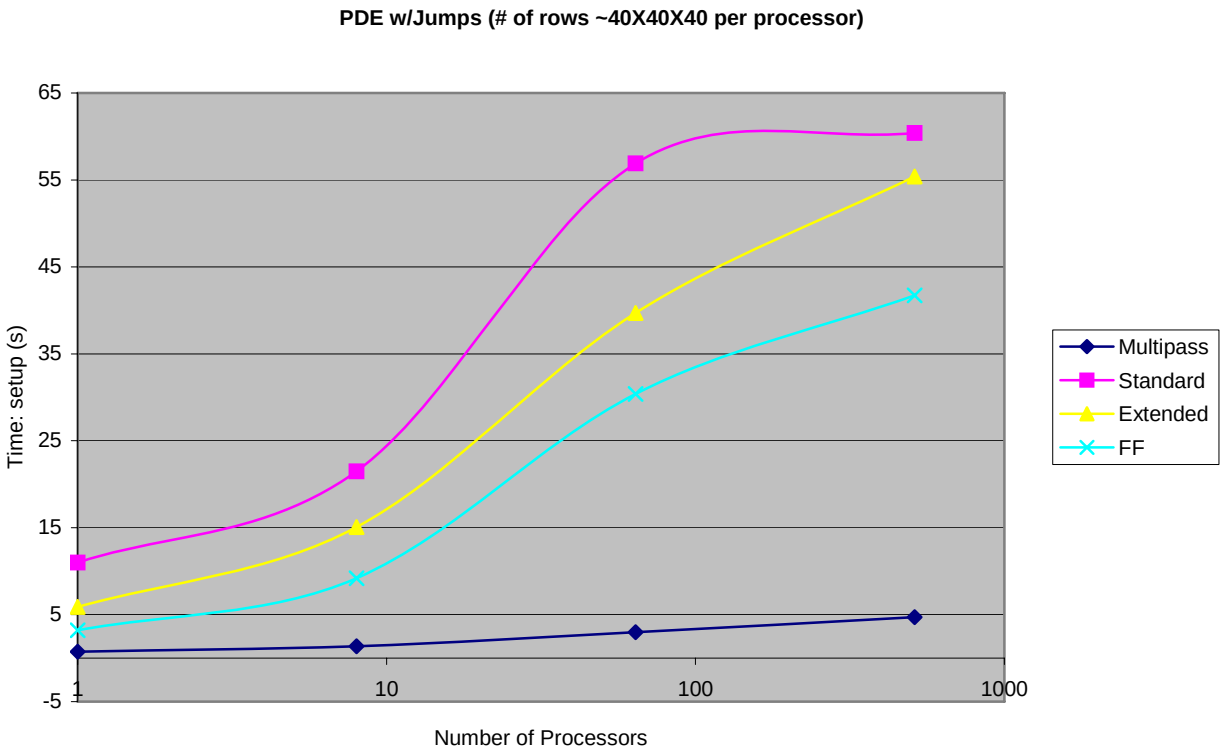

Figure 3: BoomerAMG setup times.

The final 3d case is a problem with anisotropy in the y-direction. This also shows the importance of the long range interpolation schemes.

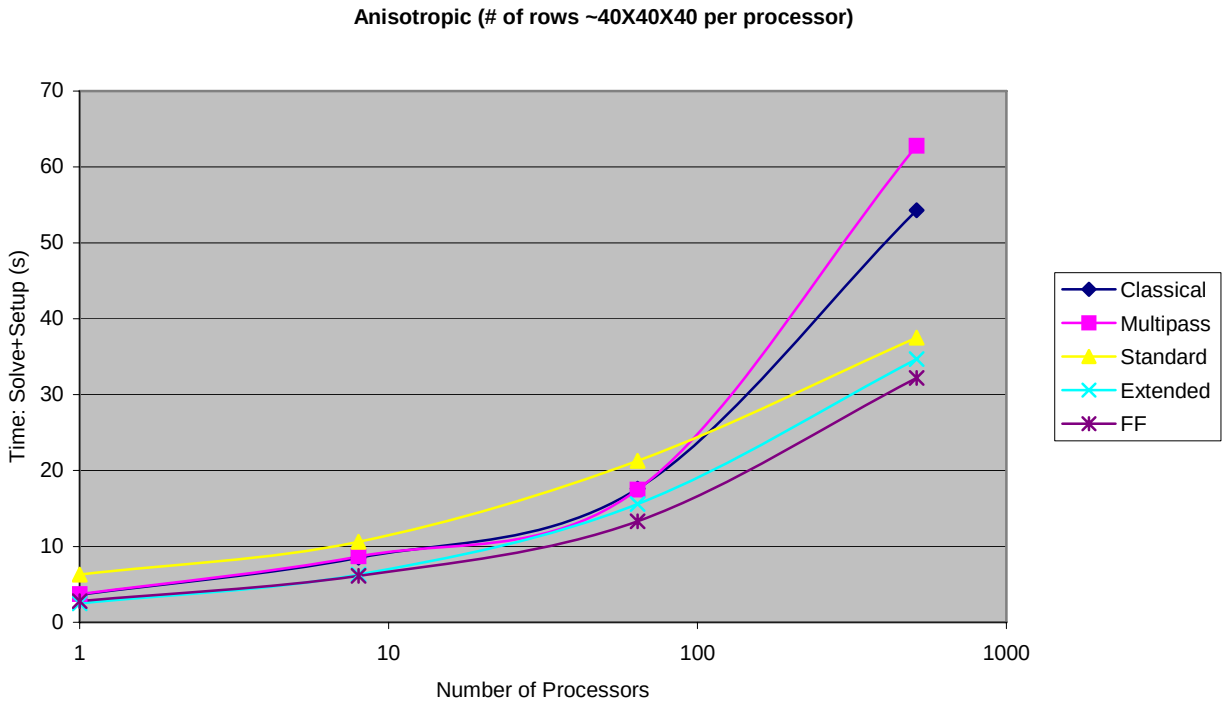

Figure 4: Time to needed to reduce the relative residual below 10^{-7}.

The classical and multipass schemes are starting to blow up as the problem size starts to get big. It takes approximately 135 iterations for the old routines to converge, compared with 15 for the long-range interpolations. All of the three new methods look to be similar for this problem. The standard and extended schemes need less iteration than F-F, but the F-F routine consistently has lower setup times by enough to be slightly quicker in the timings provided above.

The first 2d test is a 45º rotated PDE with anisotropy. As was observed in the last two 3d tests, it is necessary to use long-range interpolation.

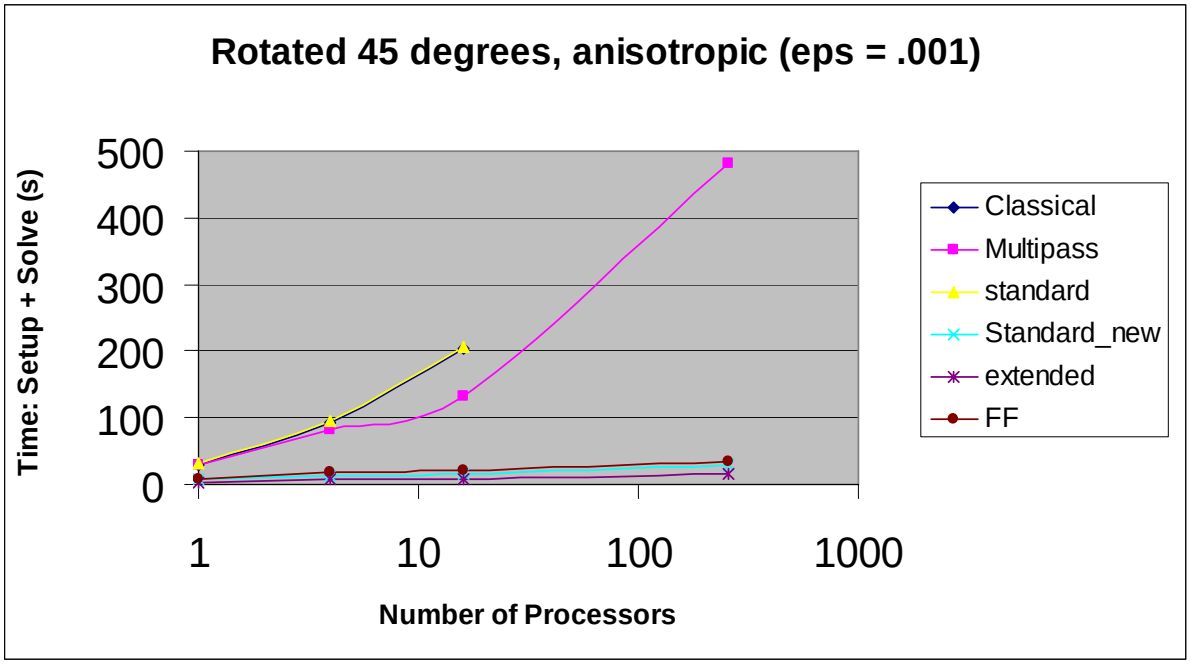

Figure 5: Time to needed to reduce the relative residual below 10^{-7}.

The original implementation of the standard interpolation routine failed to converge as the problem size grew. This is an example of a case where no separation between negative and positive connections in the weighting formula improves convergence dramatically.

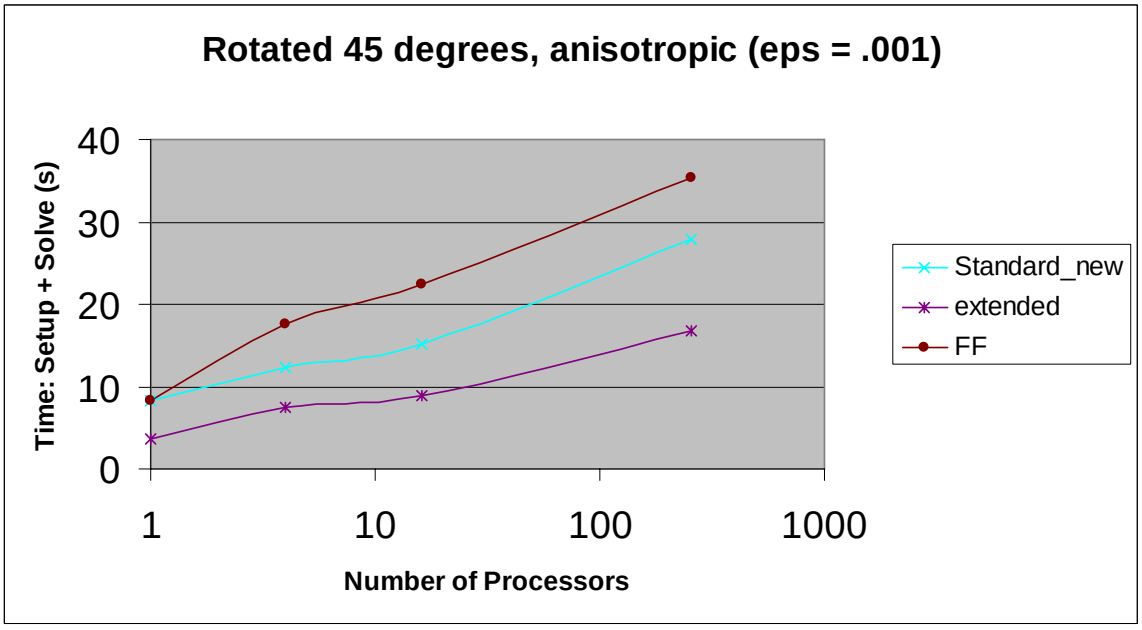

Figure 6: Time to needed to reduce the relative residual below 10^{-7}.

For this problem, the extended routine is the fastest and uses the least number of iterations, but all three of the above algorithms would be viable.

The final 2d test is a 60º rotated PDE with anisotropy. For this case, the only two methods that worked, were extended and F-F. The extended routine uses considerably less iterations to converge than F-F (15 versus 46 for 256 processors), which is why the time for setup plus solve is much better for extended.

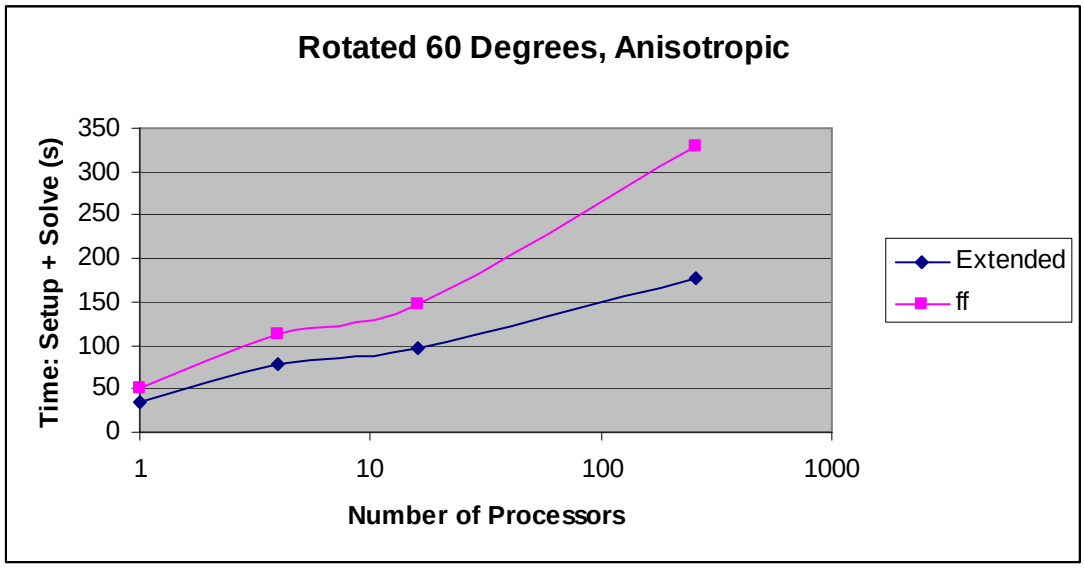

Figure 7: Time to needed to reduce the relative residual below 10^{-7}.

Conclusion

Long range interpolation was shown to be necessary for acceptable convergence in the above problems, outside of the Laplacian. Out of the long range schemes, extended and F-F were generally superior to both standard schemes. Although both extended and F-F performed similarly for the 3d case, F-F was always slightly faster. However, in the 2d test problem, extended has the best performance, and it appears to be getting better as the problem size increases. The bottom line is; for these types of problems, if PMIS coarsening is used, F-F or extended interpolations should be used as well.

References

[1] Stuben, K, “An Introduction to Algebraic Multigrid”, Multigrid, p. 413, Academic Press, 2001.

[2] Butler, Jeffrey, “Improving Coarsening and Interpolation”, Technical Report, Applied Mathematics Department, Waterloo, Ontario, Canada.

[3] Briggs, William L., Henson, Van Emden, McCormick, Steve F., A Multigrid Tutorial Second Edition, SIAM, 2000.

[4] R.D. Falgout, J.E. Jones, and U.M. Yang, “The Design and Implementation of hypre, a Library of Parallel High Performance Preconditioners”, chapter in Numerical Solution of Partial Differential Equations on Parallel Computers, A.M. Bruaset and A. Tveito, eds., Springer-Verlag, 51 (2006), pp. 267-294. UCRL-JRNL-205459.

[5] H. De Sterck, U.M. Yang, and J.J. Heys, Reducing Complexity in Parallel Algebraic Multigrid Preconditioners, SIAM J. on Matrix Analysis and Applications, 27 (2006), pp. 1019-1039. UCRL-JRNL-206780.